

Zhou, J., Feng, G., Sun, Y. and Luo, H. (2020) Intelligent Block Assignment for Blockchain Based Wireless IoT Systems. In: 54th IEEE International Conference on Communications (ICC), Dublin, Ireland, 7-11 June 2020, ISBN 9781728150901 (doi:[10.1109/ICC40277.2020.9149394](https://doi.org/10.1109/ICC40277.2020.9149394))

The material cannot be used for any other purpose without further permission of the publisher and is for private use only.

There may be differences between this version and the published version. You are advised to consult the publisher's version if you wish to cite from it.

<http://eprints.gla.ac.uk/213666/>

Deposited on 16 April 2020

Enlighten – Research publications by members of the University of
Glasgow

<http://eprints.gla.ac.uk>

Intelligent Block Assignment for Blockchain Based Wireless IoT Systems

JianHong Zhou^{*†}, Gang Feng^{*}, *IEEE senior member*, Yao Sun[†], Hongxin Luo^{*}

^{*}School of Computer and Software Engineering, Xihua University

[†]National Key Laboratory of Science and Technology on Communications,
University of Electronic Science and Technology of China
E-mail:fenggang@uestc.edu.cn

Abstract—In legacy blockchain based systems, each involved node has to store a complete blockchain to ensure the system security without any central authoritative controller. However, it is usually impossible for a wireless IoT node to store a complete blockchain, especially for those simple sensor nodes without sufficient storage and computing resources. In this paper, we propose a block assignment scheme for blockchain based wireless IoT systems with aim to tackle the blockchain storage problem. Specifically, we propose to maintain a complete blockchain by a set of IoT nodes in a collaborative way on the premise of ensuring that each node can check every transaction. On the other hand, we should save the storage space of IoT nodes to the greatest extent for saving more blocks so as to maximize the lifetime of IoT nodes. We formulate this optimal block assignment problem as a 0-1 mixed integer-programming problem. We propose to incorporate Chaotic optimized algorithm into Genetic algorithm to provide an efficient near-optimal solution. Compared with the brute-force and conventional Genetic algorithms, our proposed algorithm can achieve the minimum storage occupancy to store blocks. Meanwhile, the proposed algorithm has the lowest computational complexity.

I. INTRODUCTION

With the rapid development of IoT applications, the security of IoT system has received increasing attention [1]. As a distributed ledger technology, blockchain technology provides the infrastructure for the trusted exchange between IoT devices. The authors of [2] mentioned that the blockchain technology is a huge engine driving the rapid development of IoT, which is mainly reflected in the following three aspects: 1) The blockchain technology makes it possible for IoT being operated in decentralization infrastructure. In this case, no central server with all data exists, which effectively avoids information leakage. Meanwhile, the secure encryption technologies used in blockchain, such as asymmetric cryptographic algorithm etc., ensure the user privacy protected to the maximum extent. 2) The distributed connections of blockchain make it possible to organize IoT as a peer-to-peer (P2P) network with a huge number of nodes. The network architecture of IoT could be optimized by using the idle resources and the P2P connections to reduce the data transmission cost. 3) The smart contract programming realizes the automatic information transmission

and processing. Thus, we not only save the operational cost of central server, but also realize low cost connection among tens of thousands terminal equipments by using blockchain in IoT systems.

Although the use of blockchain technology in IoT systems has absolute advantages, there are also some challenging problems to solve. In existing blockchain based systems, no matter what kind of consensus mechanism is used, every involved node has to store a complete blockchain to ensure the system security without any central authoritative controller. Therefore, it brings a severe challenge to the IoT nodes with limited resources for the application of blockchain technology. Indeed, most of IoT nodes only have limited storage resource, and are connected by complex and unstable wireless network environment in some IoT systems. Thus, it is unable or inefficient to store a complete blockchain in every terminal devices if the blockchain is too long, which becomes a technical bottleneck for applying blockchain in IoT systems. The authors of [3] proposed a consensus unit-based storage scheme for blockchain based IoT systems. However, they assume that the devices, which are used to collaboratively store the complete blockchain within one consensus unit, are connected by wired network. Thus, any node could check any transaction by querying all the other nodes within the same CU to recover a complete blockchain in an almost error-free transmission way. However, they did not consider wireless IoT scenario. The optimal block assignment is also not addressed.

To exploit blockchain technology in wireless IoT systems to improve the system security, in this paper, we propose a block assignment scheme to resolve the aforementioned technique bottleneck. In a wireless IoT system, we cluster all nodes into different consensus units (CUs), similar with that in [4]. The nodes within a CU should maintain the complete blockchain collaboratively. Meanwhile, a node is only able to communicate with its neighbors within its transmission coverage area. Under this circumstance, an appropriate block assignment scheme should ensure that every node could check every transaction through communicating with its neighbor nodes. On the other hand, similar with the idea in [4] that if the storage is insufficient to store a whole blockchain, pruning should be applied to the historical blocks. Thus, an optimal block assignment scheme should use the limited storage of the

This work was supported by the Ministry of Education of China, Chunhui Project under Grant number 192618, the National Science Foundation of China under Grant number 61871099 and 61631004, and the Fundamental Research Funds for the Central Universities under Grant number ZYGX2019J122.

CU to store as long as possible blockchain.

Considering the problem is a typical 0-1 mixed integer-programming (MIP) problem with binary variables and the constraints of storage and wireless link, traditional convex optimization methods cannot be applicable to solve the problem [5]. To overcome premature convergence to local optimal solution of genetic algorithm, we develop an improved genetic algorithm, namely Chaotic optimized based genetic (COG) algorithm, to obtain the near-optimal block assignment solution. The result of Chaotic optimized method is employed to be the father population to avoid the genetic algorithm from falling into the local optimal solution as well as to accelerate the genetic algorithm to obtain the best near-optimal solution.

The rest of the paper is structured as follows. We describe the system model and formulate the problem in Section II and Section III respectively. Section IV presents the near-optimal block assignment (NOBA) scheme. Section V concludes the paper.

II. SYSTEM MODEL

We consider an IoT system as shown in Fig 1. We apply blockchain to ensure the data security in this IoT system, which we call BC-IoT system. We assume that all IoT devices are connected by a heterogeneous wireless network in our framework, including eMTC network, NB-IoT, LTE, and 5G NR, etc.. Similar to that in [3], we deploy consensus units (CUs) to store a complete blockchain instead of using single node in other legacy blockchain based system. Every CU consists of a group of IoT devices with limited storage capacity and they store the complete blockchain in collaborative way. All the CUs are connected with each other to form the BC-IoT system and each CU should maintain an identical blockchain by using some kind of appropriate consensus mechanism [6]. From the function point of view, an IoT node within a CU is indeed an independent “miner” and dependent “ledger”. It means that every node can compete to generate a new block and record transactions into the block individually. However, all nodes in a specific CU have to collaboratively store a complete and effective blockchain as every node only stores partial blocks of the blockchain.

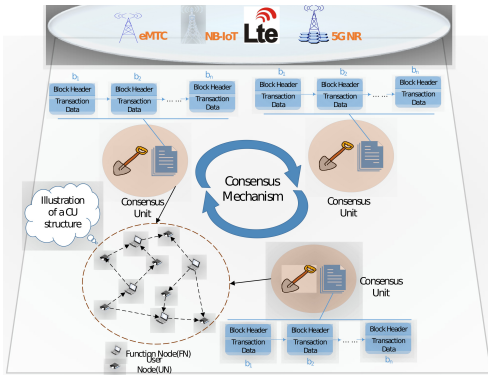


Fig. 1. BC-IoT System

Within a CU, there are two kinds of nodes, namely Function Node (FN) and User Node (UN) [7]. The FNs are the devices

which can provide blockchain related service, such as generating new blocks and storing blocks. More precisely, the blocks of a blockchain are stored in all FNs in a distributed way within a specific CU. The UNs are the devices generating transaction data and using blockchain services. We use a simple example shown in Fig 2 to illustrate the basic structure of CU. Note that a UN could only be connected with its neighboring FNs instead of all FNs included in the same CU, and an FN stores only a part of the blockchain. In this CU, UN 4 is connected with 4 neighboring FNs with wireless links. If UN 4 wants to check the transactions recorded in block 2, it should broadcast its checking request to all the connected FNs. After receiving the checking request, FN 2, FN 3 and FN 4 with the required block will send block 2 to UN 4 as the feedback. However, a wireless connection could be unstable due to the vulnerability of a wireless channel. For example, if the wireless link between UN 4 and FN 3 is too poor so that UN 4 cannot receive the correct block 2, which means a failure feedback. Although UN 4 is connected with FN 1, there is no feedback from FN 1 to UN 4 as block 2 is not stored in FN 1. In this case, UN 4 can only receive 2 copies of block 2. Based on blockchain technology, the integrity and effectiveness of block 2 can be validated by UN 4 through comparing the two received copies of block 2. Although the validation procedure is more secure if more copies of the required block can be received from multiple FNs according to blockchain technology. However, too many duplicates may reduce the life-time of the blockchain in a resource constrained IoT system. Thus, an optimal block assignment scheme should minimize the occupied storage ratio (OSR) under a certain system security requirement to store more blocks so as to maximize the blockchain life-time of the IoT system.

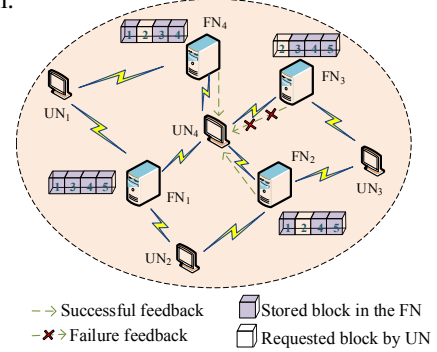


Fig. 2. The illustration of CU

III. PROBLEM FORMULATION

Based on the analysis above, we formulate the optimal block assignment scheme as follows. Assume that \mathcal{I} FNs and \mathcal{M} UNs are randomly located in a CU. $D = \{d_{mi}\}$, ($1 \leq m \leq M, 1 \leq i \leq I$) is the distance vector among UNs and FNs, and d_{mi} is the distance between UN u_m and FN f_i . Thus for u_m , the receiving SNR (Signal-to-Noise Ratio) SNR_{mi} from FN f_i is given by

$$SNR_{mi} = (Pg(d_{mi}))/\sigma, \quad (1)$$

where P represents the transmit power of FNs, $g(d_{mi})$ is the path loss between u_m and f_i , which is determined by

the specific channel model. σ is the noise power. We assume that u_m could decode the received information from f_i when SNR_{mi} is greater than a threshold β . Thus we define a binary variable as

$$[SNR_{mi}]_{\beta} = \begin{cases} 1, & SNR_{mi} \geq \beta \\ 0, & \text{otherwise} \end{cases}, \quad (2)$$

where $[SNR_{mi}]_{\beta} = 1$ indicates that the UN is connected to the FN, otherwise $[SNR_{mi}]_{\beta} = 0$.

Assume the number of blocks in the blockchain to be stored in the FNs is J . To simplify the problem, the length of a block is L MB/block. Thus, the total length of the blockchain is $BL = (L \times J)$ MB. The capacity for all the FNs is $C = \{c_i\} (1 \leq i \leq I)$. As a CU needs to store at least one complete blockchain, it has to satisfy that $BL \leq (\sum_{i=1}^I c_i)$. We assume that when UN $u_m, (1 \leq m \leq M)$ want to check the transactions recorded in block $b_j, (1 \leq j \leq J)$, it will broadcast the request to all connected FNs. We define that if the any UN $u_m, (1 \leq m \leq M)$ can receive θ identical copies of any block $b_j, (1 \leq j \leq J)$, the checking procedure is successful. The value of θ depends on the system security requirement. The greater the θ is, the higher security level the system can be achieved, and naturally more node storage capacity is consumed.

Next, we define a two dimensional matrix $\mathcal{P} = \{p_{ij}\}, (1 \leq i \leq I, 1 \leq j \leq J)$ to represent the block assign scheme, where

$$p_{ij} = \begin{cases} 0, & \text{block } b_j \text{ is not stored in FN } i \\ 1, & \text{otherwise} \end{cases}. \quad (3)$$

Let $R = I \times J$ be the number of elements in matrix \mathcal{P} . Based on the analysis mentioned above, we can design an optimal block assignment algorithm to minimize the OSR of the CU. Thus, it is reasonable to define a metric, average occupying storage ratio, to measure the degree of storage occupancy, which is defined as

$$\alpha = \frac{L \times \sum_{i=1}^I \sum_{j=1}^J p_{ij}}{\sum_{i=1}^I c_i}. \quad (4)$$

Therefore, our problem can be formulated as

$$\mathcal{P}^* = \min \alpha(p_{ij}^*) \quad (5)$$

$$s.t. \quad 0 \leq L \times \sum_{j=1}^J p_{ij}^* \leq c_i, \quad (\forall i \in [1, I]) \quad (5.1)$$

$$0 \leq \sum_{i=1}^I p_{ij} \times [SNR_{mi}]_{\beta} \leq I, \quad (\forall m \in [1, M], \forall j \in [1, J]) \quad (5.2)$$

$$p_{ij}^* \in \mathcal{P}. \quad (5.3)$$

Obviously, the block assignment problem (5) is a typical 0-1 MIP problem. Currently, there are three types of algorithms for solving such problems: precise algorithms [8] (i.e. dynamic programming, recursive method, retrospective method, branch-and-bound method, etc.), approximation algorithms [9] (i.e. greedy algorithm, Lagrange algorithm, etc.), and intelligent optimization algorithms [10] (i.e. simulated annealing algorithm, genetic algorithm, genetic annealing evolutionary algorithm, ant colony algorithm, etc). By using precise algorithms, optimal solutions can be achieved. However, the computation and time

complexity may grow exponentially with the number of variables, namely the number of FNs within a CU and the number of blocks in our problem. Thus, it is more suitable for “small scale” network, which means the number of FNs and blocks is small. Compared with precise algorithms, approximation algorithms could achieve the near-optimal solution efficiently while the accuracy of the solution is poor. As a popular algorithm, the intelligent optimization algorithm can achieve approximation solutions with high accuracy within small time complexity. Thus, it is more suitable to solve our problem in “moderate scale” and “large scale” network. Next, we design an intelligent algorithm, namely COG algorithm, to solve our problem.

IV. NEAR-OPTIMAL BLOCK ASSIGNMENT SCHEME

Based on the analysis above, we design a general algorithm to obtain the near-optimal block assignment (NOBA) scheme to minimize the OSR on the premise that every UN within the CU can check every transaction securely. Without loss of generality, we can focus on analyzing a CU of the BC-IoT system.

Chaotic motion system is a system with fine internal structure, which has the characteristics of randomness, ergodicity and regularity. Especially, ergodicity means that chaotic motion can traverse all states in a certain range without repeating according to its own laws. Thus, the chaotic optimized method based on this characteristic can assist GA to escape from local optimal solution [11]. Next, we design a chaotic genetic based (COG) algorithm to solve the NOBA problem in “large scale” network.

A. Chaotic optimized based genetic (COG) algorithm

1) *Flowchart of COG algorithm:* To solve the optimal block assignment problem, we propose to improve the genetic algorithm by embedding the solution of chaotic optimized algorithm into the genetic algorithm to overcome the premature problem of GA. We use the solution of chaotic optimized algorithm as the father population to generate the new generation. The framework of the COG algorithm is shown in Fig 3.

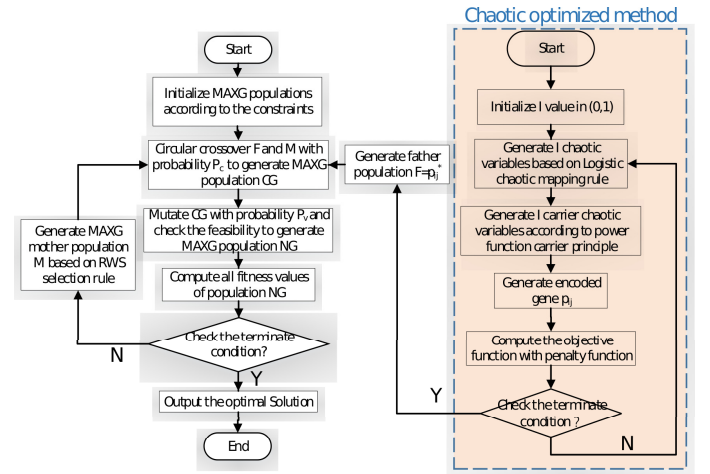


Fig. 3. The flowchart of COG algorithm

In our problem, the two-dimension matrix \mathcal{P} is a 0-1 matrix, which can be used as the encoded genes in GA algorithm. Thus, we use $\mathcal{P}_k^r, (0 < r \leq MAXP)$ to represent r_{th} gene in k_{th} generation, and \mathcal{P}_k^r has to satisfy the constraints condition of (5.1)-(5.3). $MAXP$ is the number of genes in each generation. Crossover is the core operations to improve the algorithm searching capability. We use modified chaotic searching algorithm and Roulette Wheel Selection (RWS) selection rule to generate the father and mother population respectively, to execute the circular crossover with probability $P_c, (0 < P_c < 1)$. It ensures the genetic excellence and diversity and finally accelerates the algorithm convergence. Mutation is performed to the genes to prevent the algorithm from falling into local optimal solution. We performed the mutation by changing the position of the first element 1 for every gene (every row in the matrix) with probability $P_v, (0 < P_v < 1)$. It is necessary to check the feasibility of all the mutated populations, and the new populations are expressed as NP. In the GA algorithm, fitness function is used to judge the pros and cons of the genes. For optimization problem, the gene is closer to the optimal solution, the value of the fitness function should be higher. Thus, the fitness function should be related to our objective. In our algorithm, we define $fit_k^r(t_k)$ as the fitness function of r_{th} gene in k_{th} generation.

$$fit_k^r(t_k) = \exp(-(\frac{\alpha_k(\mathcal{P}^r) - \alpha_{min}}{t_k})), \quad (6)$$

where $\alpha_k(\mathcal{P}^r)$ is the average occupied storage ratio for r_{th} input gene \mathcal{P}^r in the k_{th} generation. α_{min} is the minimum average occupied storage ratio for $MAXP$ genes in k_{th} generation NP. t_k is a temperature parameter in k_{th} generation. Next, we use RWS to generate mother populations. If the fitness value of the k_{th} generation is $FIT_k = \{fit_k^1(t_k), fit_k^2(t_k), \dots, fit_k^{MAXP}(t_k)\}$, the copied probability can be calculated as

$$P(\mathcal{P}_k^r) = \frac{fit_k^r(t_k)}{\sum_{i=1}^{MAXP} fit_k^i(t_k)} \quad (7)$$

Next, we randomly generate values $\{l_1, l_2, \dots, l_{MAXP}\}$ in $[0, 1]$. If $l_r \in [\sum_{i=1}^{r-1} P(\mathcal{P}_k^i), \sum_{i=1}^r P(\mathcal{P}_k^i)]$, \mathcal{P}_k^r will be chosen as one of the mother population in the $(k+1)_{th}$ generation.

2) *Chaotic optimized method to generate father population:* In our COG algorithm, to ensure the high quality, diversity and randomness of every generation, we design a chaotic optimized method to generate the father population. In the chaotic optimized method, power function mode is used as a carrier method to generate the modified chaotic variables. Our problem is an optimization problem in R dimension space. First, we give I different initial values to generate I chaotic variables, which are regarded as I coordinate components. Based on the characteristic of chaotic variables, chaotic mapping will be sensitive to the initial values so that there is no correlations between the I coordinate components. It could satisfy our good ergodic requirement. However, when using chaotic system to generate variables to go through the entire space, it is very hard to ensure the searching speed and accuracy. Thus, we can use some appropriate carrier method to generate chaotic variables

to improve the optimization speed and accuracy. In this paper, we first use logistic chaotic mapping method to generate the original chaotic variables as

$$z_{k+1} = \mu z_k (1 - z_k) \quad z_k \in [0, 1], k = 1, 2, \dots, I, \quad (8)$$

where μ is the control parameter of the chaotic system; z_k is the chaotic invariant set. Although these variables have ergodicity property, they are unevenly distributed within the searching space. The point probability density is large inside the interval $[0, 1]$ interior while the point probability density is small near the ends of interval $[0, 1]$. Thus, we use the power function mode to generate the modified chaotic variables as

$$z'_k = \begin{cases} z_k^\eta & z_k \in (0, m] \\ z_k & z_k \in (m, n] \\ z_k^\theta & z_k \in (n, 1] \end{cases} \quad k = 1, 2, \dots, I, \quad (9)$$

where $m < n < 1, 0 < \eta < 1$, and $\theta > 1$, z_k and z'_k are the original and modified chaotic variables respectively. Obviously, the modified chaotic variables in interval $[0, a]$ and interval $[b, 1]$ are left shifted and right shifted respectively due to $0 < \eta < 1$, and $\theta > 1$. Thus, we can increase the point probability density near the ends of interval $[0, 1]$ by adjusting the parameter η and θ . It has proved that $z'_k \in (0, 1)$, and z'_k still have ergodicity. Next, the modified chaotic variables are mapping to the ratio of occupied storage of all FNs. The number of blocks stored in the corresponding FN can be calculated as

$$NB_i = \lfloor \frac{c_i \times z'_i}{L} \rfloor, \quad i = 1, 2, \dots, I, \quad (10)$$

which is used to help generating the population matrix $p_{ij}, (0 < i < I, 0 < j < J)$. In detail, for the i_{th} FN, we arbitrarily select NB_i elements to set their values as 1, the other elements are set 0.

3) *COG algorithm:* According to the above discussions, the COG algorithm can be summarized as Algorithm 1.

V. NUMERICAL RESULTS AND DISCUSSIONS

We simulate a blockchain based IoT system to evaluate the performance of our proposed COG algorithm. For performance evaluation, we use brute-force (BF) and conventional Genetic (CG) algorithms as comparison reference. BF is an algorithm based on intuitive or empirical construction, giving a feasible solution for each instance of the combinatorial optimization problem to be solved at an acceptable cost (referring to computation time and space). However, the length of the time to obtain the optimal solution depends on the number of the feasible solutions. CG and COG are two intelligent based algorithms which could obtain near optimal solution quickly especially for problem with large feasible solutions domain. We define convergence time as the period that the algorithm could be executed to obtain the minimum average occupying storage ratio (AOSR) within a given time. Next we compare the performance of the three algorithms in the following three way. 1) the achieved AOSR changes with time under various scale

Algorithm 1 : COG algorithm

Input: $D = (M, I)$, β , J , θ , $C = c_i$, $[SNR_{mi}]_\beta$, $[SNR_{mi}]_\beta$.**Output:** p_{ij}^* , AOSR α .

```
1: Initialize parameters: MAXP;  $k = 0$ ;  $P_c$ ;  $P_v$ ;  $q = 0$ ;  $Q$ 
   ;  $fit_{max} = 0$ ;  $r = 0$ ;
2: for  $0 < r \leq MAXP$  do
3:   Initialize the first generation  $r(0) \subseteq MP(0)$  with
   satisfying the constraint condition
4:   Calculate the fit function of the first generation  $fit_r(0)$ 
5:   if  $fit_r(0) > fit_{max}(k)$  then
6:      $fit_{max}(k) = fit_r(0)$ 
7:   end if
8: end for
9: while  $q < Q$  do
10:  Use the result of CGA as  $FP(k) = FP$ 
11:   $MP(k) = NP(k)$ 
12:  for  $0 < r \leq MAXP$  do
13:    Circular crossover  $FP_r(k)$  and  $MP_r(k)$  with proba-
    bility  $P_c$  to generate  $CP_r(k+1)$ 
14:  end for
15:  for  $0 < w \leq MAXP$  do
16:    circular mutate  $CP(k+1)$  with probability  $P_v$  to
    generate  $NP_w(k+1)$ 
17:    if  $NP_w(k+1)$  satisfies the constraint condition then
18:       $w = w + 1$ 
19:    end if
20:  end for
21:  for  $0 < r \leq MAXP$  do
22:    Calculate the fit function  $fit_r(k+1)$  of  $NP_r(k+1)$ 
23:    if  $fit_r(k+1) > fit_{max}(k)$  then
24:       $fit_{max}(k+1) = fit_r(k+1)$ 
25:    else
26:       $fit_{max}(k+1) = fit_{max}(k)$ 
27:    end if
28:    if  $fit_{max}(k+1) == fit_{max}(k)$  then
29:       $q = q + 1$ 
30:    end if
31:  end for
32:   $k = k + 1$ 
33: end while
```

for the three algorithms; 2) the convergence time changes with the number of blocks and function nodes; 3) the convergence time of COG algorithm changes with the iteration number of chaotic optimized method.

A. System Simulation Settings

We use C++ to build up a simply BC-IoT system. The UNs operating transactions and FNs supporting blockchain service such as blocks storage are randomly deployed in the system. The CU consists of 5 FNs and 20 UNs. The coverage of the CU is set as a circular area with a radius of 150m, and the other parameters of the network is similar to those in [7]. We implement the BF, CG, and COG algorithms with C++ codes. All the simulation experiments are conducted on a PC with an

Intel-i5 4 core 3.2GHz processor and 8G RAM. The operating system is Win 10.

B. Numerical Results

As the number of FNs is fixed, the scale mainly depends on the number of blocks. Thus, we first compare the convergence time as a function of the number of blocks in the three algorithms, as shown in Fig 4. From the figure, we can see that the CG and COG algorithms have almost the same converge time when the number of blocks is small, say smaller than 200. Then the convergence time of CG will increase faster than that of COG as the number of blocks increases. The reason is that when the number of block is less, the number of feasible block assignment solutions for the problem is less. It is easy to traverse to find out the optimal solution in a small feasible solution domain. However, as the number of block increases, the feasible solution domain will be expanded exponentially, and the time to find out the optimal solution will increase sharply. The two intelligent related algorithms, COG and CG algorithms, could achieve the near optimal block assignment solution almost simultaneously at beginning and our proposed COG algorithm becomes faster after the number of block reaches 150. The reason is that the two algorithms adopt genetic with potential parallel comparability, which could realize simultaneous comparison among multiple individuals so as to accelerate the searching process. Moreover, we use chaotic optimized method to optimize the selection process of father generation samples, which could further accelerate the near optimal solution obtained. In addition, the COG algorithm could overcome the disadvantage that genetic algorithm falls in the local optimal solution. That is why our proposed COG algorithm is the fastest to achieve the near-optimal block assignment scheme.

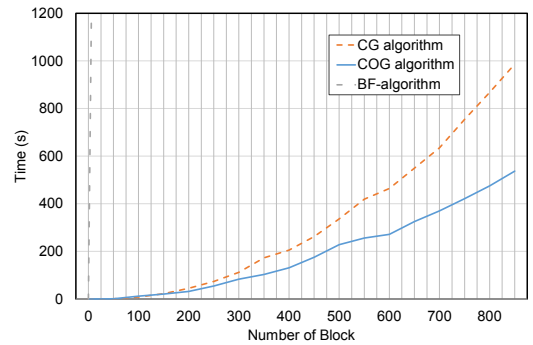


Fig. 4. The convergence time comparison for three algorithms

Next, we compare the convergency accuracy of our proposed COG algorithm as Fig 5. The BF algorithm could obtain the optimal solution after traversing all the feasible solutions under the condition that the number of blocks to save is less than 5. Thus, we set the number of FNs and blocks to save is 4 and 4 respectively, and compare the achieved AOSR among the three algorithms. Obviously, both COG and CG algorithm could obtain the target AOSR rapidly while the BF takes much longer time to traverse the whole feasible solutions to achieve the same result.

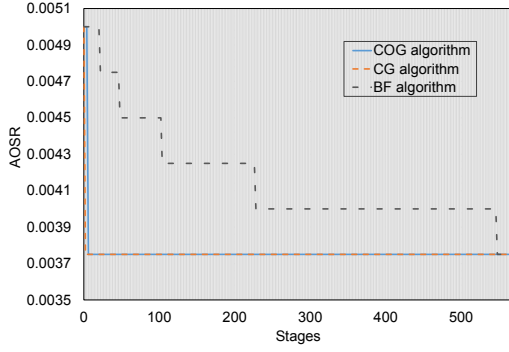


Fig. 5. Comparison of achieved AOSR for three algorithms

Next, we compare the achieved AOSR that the three algorithms could achieve within given time under the condition that the number of blocks to save is fixed as shown in Fig 6. For the sake of completeness, we conduct the simulation in the cases that the number of block to save is 400 and 800, corresponding to moderate scale and large scale respectively. Both intelligent algorithms, CG and COG algorithm, could obtain a lower AOSR while the BF algorithm can only achieve a higher AOSR within the same given time in moderate and large scales, which are more realistic. The reason is that there are too many feasible solutions for moderate and large scale scenarios, and the traversing based BF algorithm cannot find the optimal solution within the given time. For the moderate and large scale scenarios, our proposed COG algorithm is faster to achieve the same lower AOSR than the CG algorithm.

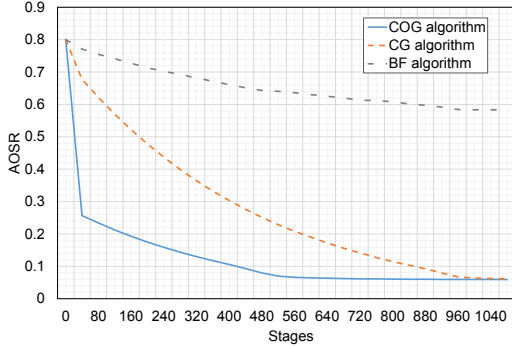


Fig. 6. The achieved AOSR comparison for large scale

Finally, we mainly focus on our proposed COG algorithm. We discuss the effect that the embedded chaotic optimized method to our proposed COG algorithm as Fig 7. The three curves differ from NC, which means that the times we execute the chaotic optimized method to generate the father generation. It shows that the more times the chaotic optimized method executed, the lower AOSR could be reached within a given time and the beginning result is better. The reason is that we embed the chaotic optimization method into the genetic algorithm to avoid trapping in local optimum and the chaotic optimized result ensures the excellence of the crossover genes for every generation. Thus, more times the chaotic optimized method is executed, the crossover genes are more excellent, which leads to that the near-optimal block assignment scheme is obtained more quickly.

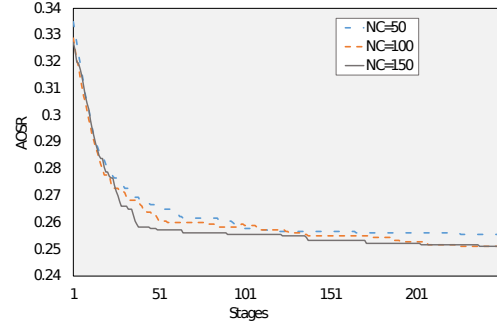


Fig. 7. The achieved AOSR comparison for various times of chaotic optimized method

VI. CONCLUSION

In this paper, we have proposed a distributed block storage scheme to solve the blockchain storage problem in wireless BC-IoT system. We first deploy CU, which consists of a set of nodes, to store a complete blockchain collaboratively. Considering the characteristics of blockchain technology, we have designed an optimal block assignment scheme, which maximizes the blockchain life-time for storage-limited wireless IoT system by reducing unnecessary storage of the block. Then we have developed an intelligent algorithm, named Chaotic optimized based genetic (COG) algorithm, to achieve the near-optimal block assignment in large scale scenarios. The simulation results show that our proposed COG can effectively find the near-optimal block assignment to enable the application of blockchain technology in the storage-limited IoT system.

REFERENCES

- [1] Office of the Deputy Assistant Secretary of the Army (Research and Technology), "Emerging Science and Technology Trends: 2016-2045," Tech. Rep., 2016.
- [2] B. Data, "2018 Blockchain and IoT industry research report," Tech. Rep., 2018.
- [3] Z. Xu, S. Han, and L. Chen, "Cub, a consensus unit-based storage scheme for blockchain system," in *2018 IEEE 34th International Conference on Data Engineering (ICDE)*. IEEE, 2018, pp. 173–184.
- [4] S. Nakamoto *et al.*, "Bitcoin: A peer-to-peer electronic cash system," 2008.
- [5] S. M. Lim and K. Y. Leong, "A brief survey on intelligent swarm-based algorithms for solving optimization problems," *Nature-inspired Methods for Stochastic, Robust and Dynamic Optimization*, p. 47, 2018.
- [6] D. Mingxiao, M. Xiaofeng, Z. Zhe, W. Xiangwei, and C. Qijun, "A review on consensus algorithm of blockchain," in *2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE, 2017, pp. 2567–2572.
- [7] Y. Sun, L. Zhang, G. Feng, B. Yang, B. Cao, and M. A. Imran, "Blockchain-enabled wireless internet of things: Performance analysis and optimal communication node deployment," *IEEE Internet of Things Journal*, 2019.
- [8] Z. Ma and S. Torquato, "Precise algorithms to compute surface correlation functions of two-phase heterogeneous media and their applications," *Physical Review E*, vol. 98, no. 1, p. 013307, 2018.
- [9] D. P. Williamson and D. B. Shmoys, *The design of approximation algorithms*. Cambridge university press, 2011.
- [10] D. Pham and D. Karaboga, *Intelligent optimisation techniques: genetic algorithms, tabu search, simulated annealing and neural networks*. Springer Science & Business Media, 2012.
- [11] X. D. Sang, X. G. Luo, Y. U. Chun-Lai, and T. Chen, "Chaos genetic algorithm for solving 0-1 integer programming problem," *Application Research of Computers*, 2011.